REMARKS/ARGUMENTS

In the Advisory Action, the final rejections of all of the pending claims 1-30 as being anticipated by Lewis (U.S. Patent No. 5,812,394) were maintained. In the Examiner's comments (on the Continuation Sheet of the Advisory Action), the Examiner indicated that the arguments presented by the Applicants in their response to the final Office Action (the Amendment faxed on February 27, 2004) were unpersuasive. Further, to the extent that the Applicants' arguments related to the absence of disclosure within Lewis of implementation of both a control logic program and a HMI program, the Examiner indicated that it was not appropriate to read these features into the claims in their current form. The Examiner did not enter the minor amendments to the claims presented in the Applicants' February 27th Amendment.

In response to the Advisory Action, the Applicants have filed herewith a Request for Continued Examination. The Request for Continued Examination requests entry of the amendments to the claims presented in the Applicants' February 27th Amendment, and further requests entry of the amendment to the claims presented herein, namely, the addition of a new independent claim 31.

Further, the Applicants are submitting with the Request for Continued Examination an Applicant Initiated Interview Request Form. As the Examiner will recall, the Applicants previously requested an interview with the Examiner on February 19, 2004 but were not granted such interview. Because the Applicants believe that an interview with the Examiner would help to clarify a number of the arguments presented by the Applicants in their several responses concerning various distinctions between Lewis and the Applicants' claims, the Applicants respectfully request that the Examiner grant this interview. The Applicants in particular would appreciate the opportunity to conduct an interview before any further Office action in this Application is taken by the Examiner.

In the present Amendment, the Applicants first discuss the new claim 31 that is being added to the present Application, as well as certain of the pending claims that relate to claim 31. The Applicants then further explain why, despite the comments made by the Examiner in the Advisory Action, the arguments presented in the earlier Amendment of February 27, 2004 are applicable even though a control logic program and HMI program are not specifically recited in independent claims 1, 16, 19, 22 and 25. Finally, for the Examiners' convenience, the Applicants repeat the arguments presented in the February

27th Amendment, so that the present Amendment document can serve as a basis for an interview with the Examiner as requested in the attached Interview Request Form.

I) New Claim 31 and Related Claims

As shown above, the Applicants have added a new independent claim 31 to the present Application. Claim 31 is largely the same as pending claim 1. However, claim 31 expressly recites that the first program fragments are control logic program fragments, that the second program fragments are HMI program fragments, and that the first and second control program portions are control logic and HMI program portions, respectively.

This claim has been added in response to the Examiner's comment in the Advisory Action that the rejected claims do not expressly recite the implementation of control logic and human machine interface (HMI) programs.

At the same time, the Applicants wish to remind the Examiner that, while none of pending independent claims 1, 16, 19, 22 and 25 expressly recite control logic and HMI programs, several of the claims depending from these claims do recite control logic and HMI programs. In particular claims 2, 17 and 24 recite control logic and HMI (or "visualization") program fragments.

Thus, to the extent that the Examiner agrees that Lewis fails to disclose the implementation of both control logic and HMI programs, the Applicants submit that not only new claim 31 but also claims 2,17 and 24 should be in condition for allowance for at least that reason.

II) Applicability of Applicants' Arguments Irrespective of Express Recitation of Control Logic and HMI Programs

The Applicants further submit that, regardless of whether the Applicants' claims expressly recite the implementation of both a "control logic" program and a "HMI" program, the Applicants arguments distinguishing Lewis from the Applicants' pending claims set forth in the Applicants' February 27th Amendment still hold true (for the convenience of the Examiner, these arguments are repeated below in part III of these Remarks).

In particular, the Applicants reiterate that Lewis does not appear to disclose the creation of a program in two stages by way of two program builders and involving two

sets of program fragments. That is, Lewis does not appear to disclose the creation of a program in two stages in which program fragments for the program are obtained from a library having multiple files each containing A and B type program fragments, where during the first stage of program creation the A type program fragments are instantiated and linked and during the second stage of program creation the B type program fragments are automatically instantiated and linked up with the A type program fragments based upon the library files in which the A and B type program fragments originated. Lewis does not, as recited for example in pending claim 1, utilize first program fragments from multiple files to form a first portion of a control program and then create a second portion of the control program from second program fragments taken from the same files of the first program.

If anything, Lewis appears to teach the creation of a single control program in a single stage. Regardless of whether Lewis may teach the use of more than one program fragment, Lewis does not appear to show two classes of program fragments where program fragments from each class are stored in each of a multiplicity of library files. In particular, Lewis does not require, and does not teach, a library that stores files that each respectively include two different types of related program components.

Further, with reference to independent claim 25 in particular, the Applicants submit that Lewis does not disclose the formation of a control program by providing a generalized map of a control system and then dragging and dropping object templates onto the map. Instead, Lewis shows the dragging and dropping of object templates only onto regions of a blank screen, where the templates must then be connected with one another as shown in FIGS. 27, 29, 31, 33 and 36, for example. Thus, in contrast to pending claim 25, Lewis fails to disclose the providing of a model having hierarchically-ordered entities each of which respectively represents at least one of a component of a system and a process of the system, and then receiving commands to associate at least some of the program fragments with the hierarchically-ordered entities.

For at least these reasons, therefore, the Applicants submit that numerous arguments presented in the February 27th Amendment (and repeated below) do not depend upon the express recitation of control logic and HMI program fragments within the Applicants' pending claims.

III) Arguments Presented in Remarks of February 27th Amendment

A) Conceptual Differences Between the Applicants' Invention and Lewis

The Applicants recognize that Lewis generally bears some relevance to the subject matter of the present Application, insofar as Lewis concerns a computerized development system for developing control schemes for facilities. Nevertheless, the Applicants respectfully submit that there are significant conceptual differences between the Applicants' invention and the system shown in Lewis. To facilitate the Examiner's understanding of these differences, the Applicants not only provide the discussion below but further include a table (Table I below) providing a comparison of various characteristics of the Applicants' invention and those of Lewis.

1) The Applicants' Invention

The Applicants' inventive system is intended to allow the development of system control programs that include multiple programming portions such as control logic and human-machine interface (HMI) programming. Additionally, the Applicants' system is intended to allow development of HMI programming after a control logic portion of the system programming has been completed, which is desirable for reasons such as those discussed in the present Application at page 3, lines 19-25. The design of the Applicants' system is further intended to make this programming as simple and automatic as possible.

To achieve these goals, the Applicants' system begins the development of a system's control programs by starting with a high-level diagram such as an "area-model" conforming to the S88 standard (see page 8, lines 6-10). This high-level diagram generally shows the relationships/linkages among components of the system for which the program is being designed, and provides names for those various components.

Next, in a first stage of development, a user configures a control logic program for the system represented by the area-model by selecting first program components concerning control logic from a library and "dragging and dropping" those components onto components of the area-model. Because the area-model itself specifies relationships among system components and also includes names for those system components, it is not necessary for a user to specify information regarding interconnections or names when associating the first program components with the area-model. Further, because the first program components in the library are "smart" in that they are highly-developed

templates, it is not necessary for the user to provide any additional configuration information. That is, aside from the dragging and dropping commands provided by the user, the creation of the control logic program is automatically created by a control program integration wizard (as discussed on page 11, line 10 through page 12, line 26).

Later, in a second stage of development subsequent to the forming of the control program for the system, the user may wish to create a human-machine interface (HMI) program for the system, where the HMI program properly interacts with the already-created control logic program. In accordance with the Applicants' invention, once the user provides a command (and possibly a name) for such an HMI program, an HMI integration wizard automatically creates the HMI program using second program components stored in the library, and properly integrates the HMI program with the preexisting control logic program (as discussed on page 13, lines 3-29).

This process of creating the HMI program and integrating it with the existing control logic program is entirely automatic and is possible in part by virtue of the manner in which the first and second program components are stored in the library. That is, in accordance with the Applicants' invention, the library includes multiple files corresponding to different possible system components, and the respective files store both first and second program components (that is, program components of different types such as control logic and HMI) together when those program components are related to each other in terms of the system component to which they pertain. Because the library stores related program components of the different types together in files corresponding to particular system components, the HMI integration wizard is able to select second (HMI) program components that properly correspond to each of the first (control logic) program components that are already implemented in the control logic program, and able to integrate those second program components properly with the first program components.

2) Differences between Lewis and the Applicants' Invention

Although Lewis relates generally to creating control programming for a system, it differs in numerous respects from the Applicants' invention. To begin with, Lewis does not have the same goals as the Applicants' invention. In particular, while Lewis may refer generally to graphical-type information as being one of the possible components of the templates used to create control programs (e.g., at col., 28, lines 38-39), Lewis fails to

indicate that it would be desirable to create a HMI program for a system after a control logic program for the system had already been created, and entirely fails to disclose such two-stage creation of two program portions. If anything, Lewis appears to teach the creation of a single control program in a single stage. Consequently, Lewis does not require, and does not teach, a library that stores files that respectively include two different types of related program components, one of which is appropriate for implementation into a control logic program during the first stage of program creation and the other of which is appropriate for implementation into a HMI program during the second stage of program creation.

Further, Lewis is not directed towards automating and simplifying the development of control programs in the same manner as the Applicants' invention. While Lewis discloses using templates from a library (e.g., at col. 27, lines 43-45), these templates are not highly-developed. Consequently, according to Lewis, a user must enter large amounts of configuration data to render those templates sufficient in detail so as to be implementable in a control program (as shown, for example, with respect to FIG. 37). Further, according to Lewis, a user must enter name information when implementing templates (see, e.g., col. 28, line 35) as well as specify linkages between implemented templates (see, e.g., col. 99, lines 17-19 and FIG. 36). Thus, the user involvement required by Lewis in creating a control program is much greater, and much more complicated, than that required by the Applicants' invention, and consequently the system of Lewis is much less automatic than the Applicants' invention.

That Lewis is much less automatic than the Applicants' invention is due in large part to the fact that Lewis does not employ a process in which an area-model or other high-level system model is created prior to implementing templates and otherwise creating the control program. That is, while the creation of control programs using the Applicants' invention involves dragging and dropping specific template information onto a generalized "map" of the system, Lewis instead merely shows dragging and dropping templates onto a blank slate. This can be seen by viewing, in succession, FIGS. 27, 29, 31, 33 and 36 of Lewis, which show how a control program is "built-up" from nothing using the templates. Because Lewis does not employ an area-model, information concerning template names and interconnections among templates must be provided piece-by-piece as the user selects and implements templates, instead of being automatically provided by virtue of the area-model as in the Applicants' invention.

To summarize, therefore, the Applicants' invention differs from Lewis on multiple levels. While the Applicants' invention employs a two-stage process for developing two different types of program portions using a library employing files with two types of programming components, Lewis discloses only a single-stage process and a library in which all program components are lumped together. While the Applicants' invention employs a development process that is facilitated by the use of a generalized, high-level model of the system, with respect to which a user associates templates from the library, Lewis discloses only the simultaneous creation of an overall system model and control program. And while the Applicants' invention reduces the amount of specific information that a user must enter when designing a control program, through the use of "smart" templates and the high-level system model, Lewis requires a much higher degree of user interaction when creating a control program and consequently provides a much more complicated, less automatic process than the Applicants' invention.

3) Table I

Applicants' System	Lewis
Begin with high-level "area-model" showing system components' interrelationships and names.	Begin with blank screen.
Identify and implement program components corresponding to system components from library in two stages, e.g., one for control logic and one for HMI programming.	Identify program components corresponding to system components from library in a single stage, where all program information for a given system component is selected and specified.
Library includes multiple files each having multiple related program components of different types, e.g., each file has a first program component corresponding to control logic and a second program component for HMI programming.	No indication that a library is provided that has files organized in a manner that involves keeping together, in a single file, program components of different, distinct types that are related to the same system component.
During first stage of development, simply drag and drop "smart" program components onto components of area-model.	There is no area-model or other high-level model on which program components can be dropped; rather, a user places program components onto blank regions of screen.
Because program components are "smart", no configuration information needs to be added.	Large amounts of configuration information must be provided to configure the "templates" obtained from the library.
Because program components are dropped onto area-model that already specifies names of system components, naming information need not be provided by user when dropping program components onto area-model.	Names must be provided to the templates when they are placed onto the blank regions of the screen.
Because program components are dropped onto area-model that already specifies interrelationships among system components, appropriate links among program components are automatically created when dropped onto area-model.	Because there is no area-model or other high-level model on which program components can be dropped, the user must also specify linkages between program components when placing those components onto blank screen regions.
During second stage of development, second program components (e.g., concerning HMI) are automatically obtained from library based upon first program components (e.g., concerning control logic) already implemented during first stage of development.	There is no second stage of development; there is no automatic retrieval of program components from a library based upon already-implemented program components.
During second stage of development, second program components are automatically configured for proper interaction with first program components already implemented during first stage of development.	There is no second stage of development; there is no automatic configuration of program components since all appropriate program components were already incorporated into the overall system control programming.

B) Allowability of Pending Claims 1-30 Over Lewis

The Applicants further submit that each of the pending claims 1-30 recites one or more of the aforementioned characteristics of the Applicants' invention that differ from Lewis, and consequently that all of the pending claims 1-30 should be allowed over Lewis.

1) Allowability of Independent Claims 1-18

a) Library Mgr. As Recited in Claims 1-18 Missing From Lewis

Although Lewis discloses using a library of templates, the Applicants respectfully submit that Lewis fails to disclose a library manager as recited in independent claims 1 and 16. The Applicants' claimed library manager is designed specifically for operation in conjunction with the Applicants' overall system, which as discussed above creates programs in two separate stages. Consequently, the Applicants' claimed library manager collects "in unique files, at least first and second program fragments having shared control variables determining physical inputs or outputs exchanged with the industrial process". Because second program fragments are stored in the same respective files as first program fragments with which those second program fragments are related, the Applicants' system is capable of automatically building the second (HMI) program during the second stage of operation based upon the first (control logic) program that has already been created during the first stage of operation.

This operation of the library manager is entirely different from that shown in Lewis since, as best as the Applicants can determine, Lewis fails to show a library that has unique files in which are stored control logic program fragments together with HMI program fragments that relate to those control logic program fragments in terms of the system component to which they relate. While the excerpt from Lewis referred to by the Examiner appears to suggest that UCOS templates can include both "user-modifiable logic" and "graphic symbol dynamics", the Applicants are unable to find any indication that such templates are actually broken into separate program fragments, where a control logic fragment can be selected for implementation during a first stage of programming and a HMI program fragment can be selected for implementation during a second, later stage of programming. That is, the Applicants are unable to find any indication in Lewis that the library manager of Lewis facilitates independent selection and processing of related program fragments during two distinct stages of programming development.

Therefore, because Lewis fails to disclose a library manager as recited in claims 1 and 16 of the present Application, the Applicants submit that claims 1-18 are allowable over Lewis for at least that reason.

b) Lewis Fails to Disclose Both First & Second Program Builders

Further in reference to independent claims 1 and 16, each of these claims recites both a first program builder and a second program builder that respectively create first and second control program portions. As discussed above, Lewis fails to show a development program that specifically operates in two stages to create two different control program portions (e.g., a control logic portion and a HMI portion), but instead shows the creation of a control program by way of only a single stage of operation. That is, while the Applicants' invention constructs a control logic portion of the control program first, and then later constructs an HMI control logic portion of the control program, it appears that Lewis only creates a single control program all at once. Because Lewis does not create a program in two stages, the Applicants submit that Lewis also fails to show both first and second program builders as recited in each of claims 1 and 16 for performing programming in two stages.

The Applicants respectfully request that the Examiner reconsider this point in relation to the portions of Lewis that were cited in the Office action as supposedly showing first and second program builders, namely, the "device logic developing component" and the "device diagramming component". The Applicants do not recognize any resemblance between the device diagramming component and the second program builder of claims 1 and 16. And, while the device diagramming component bears some resemblance to the first program builder insofar as it involves "dragging and dropping" of program components onto a screen, the operation of the device diagramming component nevertheless differs significantly from the first program builder.

As discussed above, the process set forth by Lewis differs from the Applicants' process in that, while templates are associated with an earlier-developed area-model using the Applicants' process, templates are dropped onto blank regions of a screen in Lewis. Consequently, the operation of the device diagramming component is necessarily different from that of any process performed by any portion of the Applicants' invention. For example, because the Applicants' inventive system begins with a high-level system model that provides names for system components and information regarding how the

system components are interrelated/linked, templates can simply be dropped onto the model without any input from a user as to their names/interconnections. In contrast, Lewis requires that a user provide such name/interconnection information. Thus, despite any superficial resemblance to one or the other of the first and second program builders, the device diagramming component does not in fact operate in the same manner as either of the first and second program builders.

Further, regardless of whether the device diagramming component relates to either of the first and second program builders recited in claims 1 and 16, the device logic developing component appears not to relate to either of the first and second program builders. Rather, the device logic developing component appears to constitute merely an interface by which a user provides configuration information in relation to templates. As discussed above, this is something that the Applicants' invention does not require insofar as the Applicants' invention employs "smart" templates and thus does not require a user to provide configuration information.

Thus, the Applicants respectfully submit that Lewis fails to disclose both a first program builder and a second program builder as recited by claims 1 and 16 of the present Application and therefore, for at least this reason, claims 1-18 are allowable over Lewis.

2) Allowability of Claims 19-24 and 28-30

Similar to claims 1 and 16, independent claim 19 recites a library having two pairs of program fragments stored in two library subportions, where one of the program fragments of each pair is of a first type and another of the program fragments of each pair is of a second type. Likewise, independent claim 28 recites a library having library subportions that each store first and second program fragments. Consequently, for at least the same reasons that claims 1-18 are allowable due to their recitation of a library manager, claims 19-24 and 28-30 also are allowable.

Further, each of independent claims 19 and 28 recites that a program is created in first and second stages, where the first stage involves instantiation of first/primary program fragments and the second stage involves instantiation of second/secondary program fragments. As discussed above, the system set forth in Lewis does not build a control program in two stages (where, for example, the first relates to control logic and the second relates to HMI). Rather, to the extent that the system in Lewis can be utilized

to build a control program having both logic and graphical portions, the system is utilized to build such control program all at one time.

Although the Examiner refers to column 12, lines 36-52 and column 28, lines 31-46 as showing the building of a control program in two stages, the Applicants submit that these portions of Lewis in fact do not show such a process. In particular, while the first passage refers to a "mode for selecting device symbols" and a "mode for interrelating in a graphical manner the selected device symbols", these two modes are not the same as the first and second stages of program building performed by the Applicants' system, since (if anything) these two modes relate to the implementation of the same program fragments, not two distinct sets of program fragments. Again, therefore, Lewis shows the building of a control program in a single stage rather than in two stages.

For at least these reasons, therefore, the Applicants submit that claims 19-24 and 28-30 are allowable over Lewis.

3) Allowability of Claims 25-27

As for claims 25-27, independent claim 25 recites "providing a model having hierarchically-ordered entities each of which respectively represents at least one of a component of a system and a process of the system" and "receiving commands to associate at least some of the program fragments with the hierarchically-ordered entities". These portions of the claim refer to the dragging and dropping of object templates selected by a user from the library onto a pre-developed area-model, as discussed above.

Further as discussed at length above, Lewis does not disclose the formation of a control program in this manner. That is, Lewis does not disclose the formation of a control program by providing a generalized map of a control system and then dragging and dropping object templates onto the map. Instead, Lewis shows the dragging and dropping of object templates only onto regions of a blank screen, where the templates must then be connected with one another as shown in FIGS. 27, 29, 31, 33 and 36, for example.

For at least these reasons, therefore, the Applicants submit that claims 25-27 are allowable over Lewis.

* * *

Conclusion

Given the Applicants' Remarks and Amendments, the Applicants respectfully request reconsideration and allowance of the present Application.

The Applicants wish to invite the Examiner to telephone the Applicants' attorney at the number listed below if discussion with the Applicants' attorney would be of assistance to the Examiner or further the prosecution of the present Application.

Respectfully submitted,
Randall A. Havner et al.

By: _____

John T. Pienkos
Reg. No. 42,997
Quarles & Brady
411 E. Wisconsin Avc., Suite 2550
Milwaukee WI 53202-4497
(414) 277-5777

QBMke#5557968.1